COMP9313 2018s2 Project 2 (25 marks)

Problem 1 (13 pts):

Given a corpus of documents, compute the relative frequency f(w_j|w_i), i.e.,

$$f(w_j|w_i) = \frac{N(w_i, w_j)}{\sum_{w'} N(w_i, w')}$$

Here, N(., .) indicates the number of times a particular co-occurring term pair is observed in the corpus. In this problem, we consider that w_i and w_j co-occur if w_j appears after w_i in the same line.

Input:

The format of input files is like "pg100.txt". For example, the input could be:

```
The Works of Shakespeare, by William Shakespeare
Language: English
```

Output:

Please get the terms from the documents as below:

- Ignore the letter case, i.e., convert all words to lower case first.
- Ignore terms starting with non-alphabetical characters, i.e., only consider terms starting with "a" to "z".
- Use the following split function to split the documents into terms:

  split("[\\s*$&#/\"'\\,.:;?!\\[\\](){}<>~\\-_]+")

Each line of the output file is in format of "w_i w_j $f(w_j|w_i)$". The results are sorted by the first term (i.e., w_i) in ascending order (alphabetical), then sorted by the relative frequency (i.e., $f(w_j|w_i)$) in descending order, and finally sorted by the second term in ascending order. The relative frequency is of double precision. Given the above example, the output file is like:

```
by shakespeare 0.5
by william 0.5
language english 1.0
of shakespeare 0.5
of by 0.25
of william 0.25
shakespeare by 0.3333333333333333
shakespeare shakespeare 0.3333333333333333
```

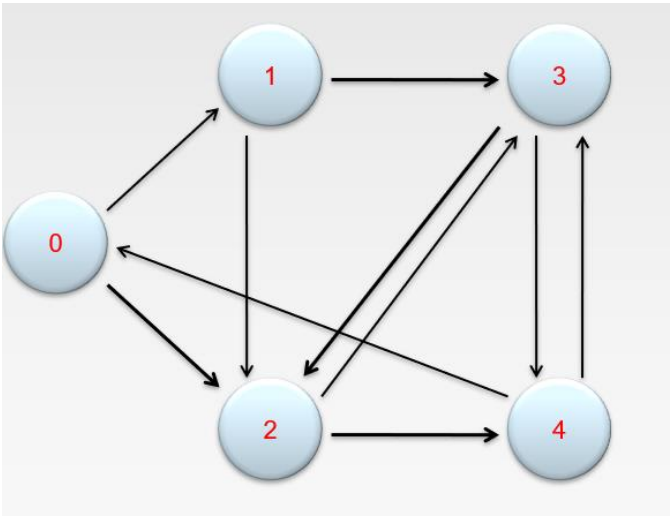
```
shakespeare william 0.3333333333333333
the shakespeare 0.3333333333333333
the by 0.16666666666666666
the of 0.16666666666666666
the william 0.16666666666666666
the works 0.16666666666666666
william shakespeare 1.0
works shakespeare 0.4
works by 0.2
works of 0.2
works william 0.2
```

Code format:

Name your package as "comp9313.proj2", your scala file as "Problem1.scala", and the object as "Problem1". Store the final result in a text file on disk. Your program should take two parameters: the input text file and the output folder.

Problem 2 (12 pts, use the GraphX pregel operator):

Given a directed graph and a number k, compute the number of cycles with length k. A cycle is a path with the same source and target node, and it does not contain smaller cycles. For example, in the below graph, 0->1->3->4->0 is a cycle with length 4, but 2->3->4->3->2 is not a cycle with length 4, since 2->3->2 and 3->4->3 are also cycles in the path.



Input:

Each line is in format of "EdgeId FromNodeId ToNodeId". Given the left graph, the input file is like below.

0 0 1 1 0 2 2 1 2 3 1 3 4 2 3 5 2 4 6 3 2 7 3 4 8 4 0 9 4 3

Output:

Show the number of cycles with length k. For example, given the sample tiny graph and 4, the result is 3: 0->1->3->4->0, 0->1->2->4->0, and 0->2->3->4->0. Note that 0->1->3->4->0 and 1->3->4->0->1 are the same cycle, and they only contribute 1 to the count.

Code format:

Name your package as "comp9313.proj2", your scala file as "Problem2.scala", and the object as "Problem2". Your program should take two parameters: a file containing the graph and a value for k. Print the result to stdout (i.e., using println()).

Documentation and code readability

Your source code will be inspected and marked based on readability and ease of understanding. The documentation (comments of the codes) in your source code is also important. Below is an indicative marking scheme:

Result correctness: 90%
Code structure, Readability, and Documentation: 10%

Submission:

Deadline: Sunday 30th Sep 10:59:59 PM

Log in any CSE server (e.g., williams or wagner), and use the give command below to submit your solutions:

$ give cs9313 project2 Problem1.scala Problem2.scala

Or you can submit through:
https://cgi.cse.unsw.edu.au/~give/Student/give.php

If you submit your assignment more than once, the last submission will replace the previous one. To prove successful submission, please take a screenshot as assignment submission instructions show and keep it by yourself.

Late submission penalty

10% reduction of your marks for the 1st day, 30% reduction/day for the following days.

Plagiarism:

The work you submit must be your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined manually.

Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct.

Do not provide or show your assignment work to any other person - apart from the teaching staff of this subject. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent.